Weekly report (2013.5.6~5.12)

Done

1) As transfer data from disk to memory is a bottle neck in my implementation, I tried to compress the intermediate data.
   Compression scheme is shown in Figure 1, structure size reduced from 28 Bytes to 12 Bytes.

Figure 1 compression scheme

old structure	improving strategy
normal：float * 3	7bit latitude + 8bit longtitude
uv：float * 2	reduce to[0.5, 1]，for each float, only keep high16bits Mantissa
z：float	unchanged
object number：uint32	64bit - normal – uv

I tested this scheme with a data block with a size of 229M, the result is shown in Figure 2.

Glossary:

[1]. opaque (rendering) : only need to read the data with the minimum depth.
transparent (rendering) : need to read all the data.

[2]. without cache : when first time reading the data, there's no cache inside the memory.
cache : there is cache after read once, so it's faster.

As we can see, cause it needs time to decompress data when handling the compressed data, it's worse at the most time, except in the case of opaque rendering and without cache.

Figure 2 experiment data

	without compression（229M）		compressed（99M）	
	opaque	transparent	opaque	transparent
without cache	7.67s	7.16s	1.69s	10.57s
cache	0.19s	0.95s	0.48s	10.3s

And cause the normal is compressed badly, to rendering quality is poor correspondingly, As shown in Figure 3.

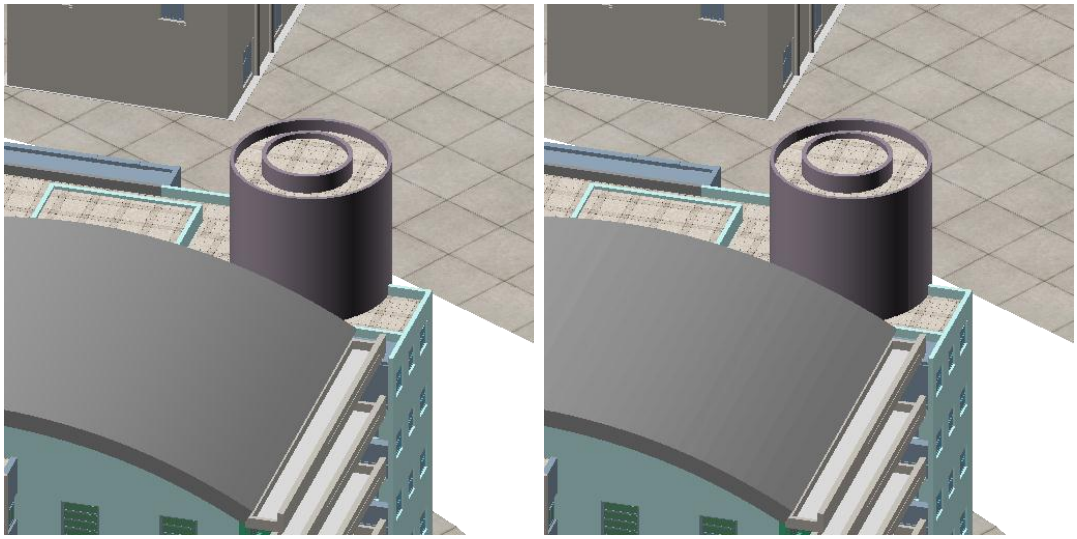

Figure 3 left without compress while right with compress

2) In order to make rendering result better, I tried the second compression scheme, shown below.

Figure 4 compression scheme2

old structure	improving strategy
normal：float * 3	reduce to[0.5, 1]，for each float, only keep high16bits Mantissa
uv：float * 2	
z：float	deleted
object number：uint32	18bit

when compressed in this way, we can get better quality as well as better compression ratio. But the program still shows worse performance when there is cache.

Figure 5 experiment data 2

	without compression（229M）		compressed（99M）	
	opaque	transparent	opaque	transparent
without cache	7.67s	7.16s	2.18s	4.93s
cache	0.19s	0.95s	0.24s	3.25s

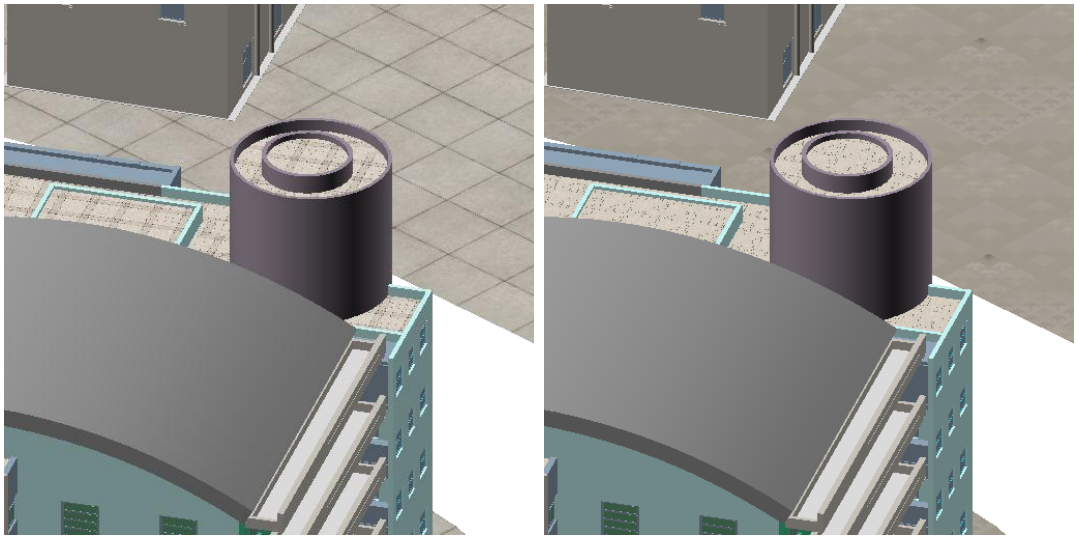

Figure 6 left without compress while right with compress (scheme 2)

3) Start reading the code of solsys and try to complete the mpi communication.

To Do

1) Continue the implement of the mpi communication.
2) Continue optimize the backend program.